TITLE OF INVENTION

Patent Application of
Christian S. Rode, U. S. citizen
2412 Stearns Hill Rd.
Waltham, Massachusetts 02451
for
Apparatus For Evaluating And Demonstrating Electronic Circuits And Components

CROSS REFERENCE TO RELATED APPLICATIONS

Not Applicable

( Provisional Application Filing: 60 / 080,905 Filing date: 4/06/98 )

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

Not Applicable

REFERENCE TO A MICROFICHE APPENDIX

A source code listing consisting of 1 microfiche and a total of 94 frames is attached.

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates to the field of computer networks and more specifically to the fields of computer networks and computer simulation.

2. Description of Prior Art

Computer programs for simulation of linear and nonlinear models have long been used to predict or evaluate the behavior of physical systems and components. These programs have also been used to educate students about linear and nonlinear system design. And these programs have previously been used in conjunction with networks (including the Internet) by means of remote terminals or display servers such as X Window.

The use of simulation programs by means of terminals or display servers has typically required a "logon account" and associated password to be created and administered for each user desiring to use the simulation software. Alternatively, guest accounts can be created for which all users share a common storage area and privileges. Both of these methods have limited applicability to public networks, such as the Internet, because

1. Individual logon accounts require time and expense to establish and maintain.
2. Individual logon accounts are not anonymous.
3. Anonymous accounts typically provide only a common storage area where one user's data are at the mercy of other users.
4. Logon accounts are inherently more difficult to make secure than access to a "web server".
5. The display of graphics typically requires the installation of a program such as an X Window server, with associated purchase, storage and maintenance costs.
6. Terminal and display server programs typically will not work across Firewalls without the installation of a special Proxy Server, with associated purchase, storage and maintenance costs.
7. Typical simulation programs do not log their usage.
8. The HTTP protocol and built-in functions of "web browsers" and "web servers" provide no support for resource management of a public CAD tool.

The following books / documents provide relevant background and are incorporated by reference:

"CGI Programming on the World Wide Web", Shishir Gundavaram, © 1996 O'Reilly & Associates, Inc.
"The Essential Client/Server Survival Guide, Second Edition", Orfali, Harkey and Edwards, © 1996 John Wiley and Sons
"Programming Perl", Larry Wall & Randal L Schwartz, © 1991 O'Reilly & Associates, Inc.
"Java in a Nutshell, A Desktop Quick Reference for Java Programmers", David Flanagan, © 1996 O'Reilly & Associates, Inc.
"HTML: The Definitive Guide", Musciano & Kennedy, © 1996 O'Reilly & Associates, Inc.
"JavaScript: The Definitive Guide, 2nd edition", David Flanagan, © 1996-7 O'Reilly & Associates, Inc.
RFC 1945 (HTTP 1.0) / 2048 (HTTP 1.1) / etc., IETF (Internet Engineering Task Force)
RFC 1866 (HTML 2.0), IETF
HTML 3.2, W3C (World-Wide Web Consortium)

As used in this document, the following words with capitalized first letters have special meanings:

A Computer includes any number and organization (cluster, array, etc.) of CPUs, memory / storage / communication devices, etc. and whose function includes the processing of data.
A Client is a Computer that is capable of accepting input from and providing output to an operator. A Client may also be a Server.
A Server is a Computer that provides computational, data storage, communication or other services for at least one (and usually more than one) Client. A Server may also be a Client.

A User is an individual or process that uses a Client. One Client can have multiple Users.

A Network includes all proxy servers, gateways, routers, communication channels, cabling, etc. that comprise a communication medium between two Computers, such as a private, local network or a public network such as the internet.

A Unique Identifier is a token (a collection of letters, digits and other symbols) that with high probability (>99%) is uniquely associated with a single User. For example, a uniformly-chosen 64-bit random number is considered a Unique Identifier for the purposes of this definition, because the probability is very small (<1%) that two users could be assigned the same number. A Unique Identifier does not necessarily contain or point to personal information about the user, i.e., an anonymous user can be assigned a Unique Identifier.

A Browser is a Client program that at least a) accepts data in the form of a display list (e.g. HTML, XML) and b) wherein at least one of the interpretable display list elements is a "hyperlink" having the capability to "link" to display list data on Servers other than (and in addition to) that which provide the list containing the link. c) uses an intrinsically stateless, file-oriented protocol (e.g. HTTP) to retrieve objects named in the display list (e.g. GIF, JPG). Typical Browsers have many other capabilities in addition to these. The words "link" and "hyperlink" are standard terms of art within the field of HTML, HTTP and the WWW.

Form Structure Data are those elements of a fetched Browser display list (e.g. <FORM> tag and associated elements) that create corresponding form elements in which data can be entered and submitted to a Server (such submitted data is Form Data)

## BRIEF SUMMARY OF THE INVENTION

The growing availability of HTTP-compatible software such as "web browsers" and Java™ makes desirable the use of the HTTP protocol as the basis for new simulation methods and apparatus.

It is therefore an object of the present invention to provide methods that enable 1) the serving of (optionally customized) Form Structure Data (in HTML, Java, etc.) to a Client, 2) the combination of Client-submitted data from said form with template or other data, 3) the processing of the combined data via a program such as SPICE and 4) the conversion and return of any processed output data to a client computer. Any and all of these steps will be referred to as simulation services.

It is another object of the present invention to provide methods that make simulation services available to an arbitrary number of simultaneous users by generating and using a Unique Identifier to keep user data separate and inhibit users from accessing each other's data.

It is yet another object of the present invention to provide methods that restrict the availability of simulation services by assigning verifiable (i.e., difficult to forge) Unique Identifiers. Such identifiers may be valid for only a limited time, in order to inhibit sharing, publishing, etc. of valid identifiers.

It is yet another object of the present invention to provide methods that limit use of simulation services to those submitted from a page created or retrieved from a particular server.

It is another object of the present invention to provide a plurality of output modes so as to be compatible with a variety of web browsers.

It is yet another object of the present invention to limit the computational resources used by any user on a per-simulation and per-time (such as day, week, month, etc.) basis.

It is yet another object of the present invention to log client submissions for marketing research, debugging, or other purposes. The log may use just the Unique Identifier and keep detailed client information in a separate file or database.

## BRIEF DESCRIPTION OF THE DRAWINGS

| | |
|---|---|
| Fig. 1 | A minimal system configuration consisting of 1 or more Clients, 1 combination web / simulation Server plus a Network. |
| Fig. 2 | Minimum Standard configuration: 1 or more Clients, 1 (or more) web Server(s), 1 simulation Server, plus Network Means |
| Fig. 3-A | High Performance Configuration (Method 1): 1 or more Clients, 1 (or more) web Server(s), more than 1 simulation Servers, plus Network Means. Simulation data submitted directly to simulation server. |
| Fig. 3-B | High Performance Configuration (Method 2): Same hardware as in 3-A, form data submitted to web server which forwards client to simulation server. |
| Fig. 3-C | High Performance Configuration (Method 3: Proxy service): Same hardware as in 3-A and 3-B, form data is submitted to web server which forwards the data to a simulation server and returns any output to client. |
| Fig. 4-A | Sample User Interface: Example login (HTML input form) |
| Fig. 4-B | Sample User1 Interface: Select Circuit Topology (HTML input form) |
| Fig. 4-C | Sample User Interface: Edit Component Values (HTML input form) |
| Fig. 4-D | Sample User Interface: Edit Additional Component Values (HTML input form) |
| Fig. 4-E | Sample User Interface: Simulation in Progress (HTML) |

## DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

The goal was to design a simple system capable of evaluating and comparing components and circuits (connected components) for the purposes of sales, marketing, educational and technical support, accessible world-wide. In all scenarios a primary advantage of the invention is to make such evaluations and comparison easier and faster, relative to the prior art. Previously, detailed component evaluation typically involved downloading and installing models and programs, with attendant risk and inconvenience to a (prospective) customer or student. The current invention also extends the advantages of a client-server implementation with respect to unburdening client computer resources (computational, storage, etc.) to anonymous users on the world-wide web (WWW) / public internet.

A secondary goal was to give unsupervised users access to simulation tools utilizing proprietary engineering data (for example, SPICE models) without giving access to the proprietary data itself. For example, the current state of the art is to distribute either 1) SPICE "macro" models (i.e. simplified models) of limited accuracy or 2) to distribute very detailed models (often under non-disclosure agreements) to customers who may be able to use the greater detail to discover proprietary design information. The current invention makes it possible to allow users to simulate real circuits with the most detailed models over the internet while keeping some model data hidden.

The preferred embodiments use HTML or Java (or both) and HTTP / CGI to create public internet / WWW interfaces to the simulation tools. There are significant technological differences between a Browser CAD tool interface and client-server CAD implementations that have come before. These differences and the requirements of public access require additional innovations to accomplish the goal as set forth above.

Some of these differences include: 1) the need to synthesize a Unique Identifier to create user state, separate user data and manage user resource use 2) preventing or automatically detecting and compensating for abuse; limiting access; load management and reporting 3) usage logging for marketing or quality control purposes, 4) the variable capabilities and compatibilities of browsers.

1. Description

Three topologies and five distinct systems are illustrated in Fig. 1- Fig. 3C. All consist of Client(s), Server(s), a connecting network and new Server methods. The distinctions among them are the number and types of Servers and the methods used therein. In the preferred embodiments these methods are implemented as Apache Unix (Linux) CGI-scripts, predominantly written in Perl. Other languages and operating systems are contemplated, for example, Java servlets running within Microsoft's Internet Information Server. In all cases, the connection network (200) is undefined and may include an unknown number of proxy servers, gateways, routers, communications channels, cabling and the like. An example, however, would be the public internet / WWW.

At least some of following known methods are distributed among the different kinds of Servers:

a) HTTP GET, POST, etc. and CGI methods (standard web server methods)
b) Methods for restricting access to a web site by a particular IP address, domain or sub-domain.

At least some of the following new methods are distributed among the different kinds of Servers:

a) A Unique Identifier assignment method.
b) An optional Unique Identifier validation method.
c) A form structure data generation method. This may include merging the Unique Identifier into a template form.
d) A form structure data and circuit synthesis method. A schematic with a corresponding component form is synthesized for user input and according to user specifications.
e) A method for merging submitted form data with other data in preparation for processing.
f) Processing merged data to produce output data compatible with a Browser. This is possibly divided into the sub-methods of creating temporary output data, then converting said data to a Browser-usable format.

g) Methods for limiting the number of simulations per Unique Identifier, per day (or other unit of time)

h) Methods for canceling simulations in progress in favor of new submissions submitted by the same user (as identified by the unique identifier)

i) Optional methods for gradually reducing the priority of successive jobs submitted by the same user.

2. Operation

In the preferred embodiments the Server methods are based on standard HTTP / CGI protocols and consequently many of the error or other unusual conditions handled by those protocols are not shown, only a typical transaction is illustrated.

In Fig. 2, the "minimum standard" Client-Server arrangement is shown: one or more Clients (100), one "standard" or "ordinary" web Server (400) and one web-based simulation Server (500). In this preferred embodiment both of these Servers have a web server, (e.g. Apache) running on each and new methods are implemented using CGI scripts. In this configuration, the Client requests HTML, image and other web data, including "form structure data" (data that comprises a list of form elements), from the standard web server, much as is the typical case for most servers on the WWW. Form structure data pertaining to simulation, however, is so constructed as to refer to the simulation Server. When the user submits data from these forms, the action is completed by the simulation Server, rather than the standard Server. This is done so as to unburden the standard Server of the computationally intense task of simulation.

In Fig. 1, the standard and simulation servers are combined into a single server, of lower cost and performance (300).

In Fig. 3A, a higher performance system is envisaged in which multiple simulation servers are available to provide a corresponding multiplication in the number of simultaneous simulations that can be processed. (The alternative of a single, multiple CPU server is considered to fall under either Fig. 1 or Fig. 2). In this case the simulation server is chosen ahead of time by the standard Server, by an unspecified algorithm, such as round-robin allocation. The server to be used is dynamically encoded in the form structure data that is sent to the client.

In Fig. 3B, a system similar to 3A is used except that the standard Server directs all simulation form submissions to itself then redirects those submissions (e.g. using the Location: directive) to one of the simulations Servers, by an unspecified algorithm.

In Fig. 3C, a similar system to 3B is used except that the standard Server directs all simulation form submissions to itself and then makes a request of at least one of the simulation servers, the response(s) to which it returns to the Client. In this configuration it is not necessary for the communication between the standard Server and the

simulation Server(s) to be HTTP / CGI and the standard Server is a kind of proxy server / gateway. Any high-performance protocol is sufficient since the Client interface is solely with the standard Server.

Figs. 4A-L show a typical Client presentation, to be used in conjunction with Fig. 2 to explain a typical simulation session.

In generating a Unique ID, it is desirable to make the creation of the ID somewhat burdensome so that replacing exhausted IDs is non-trivial. This can be accomplished by means of a ID creation screen (as shown in 4A, a login screen) which may involve the entering of subscription data, taking a survey or simply filling out a make-work form. In the preferred embodiment the ID can be inserted into the URL so that it becomes part of PATH-INFO and preserved in a cookie (HTTP / CGI terms of art). The cookie / path-info mechanism are well known.

After the ID has been generated the user's Browser is forwarded to a simulation page. As shown here, the first CGI script executing is called "get" which treats everything after "get" in the URL as PATH-INFO. The first piece of the PATH-INFO is the number 891835080547, which in this particular case is the Unique ID. At time of creation, the Unique ID is also sent as a cookie, so that if the Browser is cookie-capable the ID will be remembered automatically when the user visits the page in 4A again and they will be forwarded automatically to this page. The rest of the path-info is used to fetch other web data, in this case a Perl script.

In Fig. 4C, the user has decided to select a different circuit to simulate and is entering data into a(n HTML) form element. Fig. 4D shows some of the other options possible in this form. In 4E they have clicked the "Simulate" form submission button and a new page has popped up with a temporary message (in Netscape browsers only) that says "Simulation in Progress". In the "Fig. 2" preferred embodiment, the clicking of the "Simulate" button sent form data directly to a simulation Server as designated in the ACTION field of the <FORM> tag. The following steps are performed at this time by the simulation server.

a) The Unique ID is retrieved and checked to make sure it is authentic. If not the user Browser is directed to an error page.
b) If authentic, the Unique ID is looked up in a database (in the source listing, this is simply a text file)

to determine which account directory should be used. A simulation counter and timestamp are also retrieved and it is determined whether too many simulations have been run by this ID within the prescribed time. If too many have been run, the user Browser is directed to an error page.

c) A check is run to find existing processes running with this Unique Identifier. This may involve either a "process ID" temporary file or a search of a system process list or both. If any processes are found, they are aborted (killed).

d) The user-submitted form data is merged with a template file and the result is submitted to a simulation program such as SPICE. In some cases, the output data must be further converted to a standard format such as GIF or PDF before return. Several temporary files are typically created that use the Unique ID in their names to guarantee they don't conflict with the temporary files of other users. Part of the submitted form data determines the output format and thus the present invention includes multiple user-selected output modes for maximum Browser compatibility.

e) If the simulation is successful, the output data is returned to the user Browser (by using web server methods. If it is not successful an error page is shown.

In Fig. 4F a successful transient simulation is shown with the output as a 1024x1024 pixel GIF plot shrunk into a 512x512 area. By clicking on the GIF image, the entire 1024x1024 plot is displayed. Figs. 4G and 4H show the simulation process repeated for a frequency plot in Adobe™ Acrobat™ PDF™ format. Figs. 4I and 4J show the process repeated for a Smith chart impedance plot with the output a Java applet page that draws a display list. A URL reference to a display list is incorporated in the <PARAM>s to the Java applet is separately fetched from the simulation Server and rendered. Fig. 4K shows the ability of this Java applet to show or hide waveforms.

Fig. 4L shows that the HTML interface can be reproduced (and extended) by using Java. Moreover, the HTTP methods used by the simulation servers are available to Java, so the new Server methods are usable by both HTTP and Java interfaces, simultaneously. In this case, however, it's important to note the value of the Unique ID being available in both the PATH-INFO and a cookie, since it is believed to have been experimentally shown that a) Java applets should be given IP addresses for their CODEBASE parameter, so as to avoid a Netscape proxy server incompatibility. b) an IP address and corresponding domain name constitute separate "zones" for cookie data and so said data won't be transmitted for Java submissions to a simulation Server by an IP address.

Fig. 5A and 5B show calculator-type "assistants" or "accessories". In this case they are transmission line parameter calculators that translate physical geometry into more useful parameters. By means of JavaScript it is possible to calculate derived quantities and set those values automatically in the simulation form. Similarly, it is possible for a Java calculator to find and send messages to a Java simulation interface. Some browsers also have the capability to have JavaScript call Java and vice versa.

In Fig. 6A through 6F is presented a synthesis extension to the new Server methods previously discussed. In Fig. 6A a Java interface is presented. In Fig. 6B the synthesis function is selected (high-pass filter). In 6C other parameters have been selected and the "Calculate" button is about to be pressed. When calculate is pressed, the following steps will take place:

a) Form data is submitted to a new set of methods (a new CGI script) which synthesize the topology and component values according to the submitted data.
b) The synthesized topology and component values are converted into schematic graphical data and Form Structure Data. The synthesized Form Structure Data is similar to the generated Form Structure Data of earlier examples, except that said Data contains a new, hidden description of the synthesized circuit, or alternatively a pointer to a temporary circuit template file, synthesized in parallel with the Form Structure Data.
c) The Form Structure Data is returned to the Client. In this embodiment, the data is displayed in a separate HTML frame of the user's Browser.

At this point, the user's options are similar to those earlier when a static circuit was chosen for simulation. Proposed values can be edited and then simulated or directly simulate to confirm correct synthesis. Post synthesis simulation is a powerful tool for checking synthesis accuracy and practicality.

3. Anticipated extensions

The new simulation methods could be implemented as a "plug-in" to a web server or merged completely with web server methods so as to produce a single software entity of higher performance and less costly to install and maintain.

System synthesis methods could be used to front-end multiple circuit synthesizers, each of these circuit synthesizers distinct and yet similar in broad concept to the filter synthesis module demonstrated. Said system synthesis methods could choose the circuits that would compose a system and then automatically call each of the circuit synthesizers in turn while presenting the user with design options.

There may be other uses and extensions not yet envisioned. Although the examples and discussion are based in electronic circuits, simulators such as SPICE are widely used to simulate mechanical, biological, etc. systems and therefore nothing in this application is intended to limit its scope to purely electronic devices or systems.

Thus the scope of the invention should be determined by the appended claims and their legal equivalents, rather than by the examples given.